Cesium Based Lightweight WebBIM Technology for Smart City Visualization Management

Fan Liu[1], Huijuan Zhang[1], Yonghao Hu[1], Xinqi Guo[1], Zuoteng Zhu[1], Jinyuan Jia[1](✉), and Hehua Zhu[2]

[1] School of Software Engineering, Tongji University, Shanghai, China
jyjia@tongji.edu.cn
[2] College of Civil Engineering, Tongji University, Shanghai, China

Abstract. With the in-depth application of BIM technology in urban construction, how to efficiently load and parse BIM models with tens of millions of triangular patches, and how to combine BIM with GIS system to support the management of smart city are the two key technologies that are urgently needed to break through. This paper optimizes the lightweight of BIM data and WebBIM visualization, using Cesium as the engine of WebGIS system to perform geographic information rendering, while Three.js engine is used to render WebBIM models. The successful combination of WebGIS and WebBIM gives support to many novel solutions for smart city visualization management such as BIM cutting and view point switching. It enables users to smoothly view and edit large-scale BIM models in web browsers without any third-party plugins and establish a multi-party collaboration platform with WebGIS as the core.

Keywords: WebBIM · WebGIS · Cesium · Online Web3D visualization · Lightweight processing · Smart city

1 Introduction

With the development of the AEC (architecture, engineering and construction) industry, Building Information Modeling (BIM) technology has developed rapidly and received a lot of attention. The traditional C/S architecture with desktop application as the client is gradually replaced by the B/S architecture which uses web browsers as the client. Meanwhile, the development of HTML5 provides support for the WebBIM visualization platform.

At the same time, with the vigorous promotion of smart city, the traditional geographic information technology has been injected into fresh vitality. The traditional GIS pays more attention to the external environment and geospatial information but lacks indoor information of architectural models. If rendering BIM models on the WebGIS platform, we can not only view the external geographic environment, but also the internal structure information of BIM models.

However, it is not easy to implement the "WebGIS + WebBIM" platform for the two engines are essentially different in design and working principle. In order to realize the Cesium based lightweight visualization of WebBIM models, the technical difficulties faced mainly include:

1.1 The Storage Bottleneck of Web Browsers

As the size of BIM scene increases, the complexity of the space structure grows rapidly. The total amount of BIM data may reach the city level (100 TB or more) in the near future. Generally speaking, the memory that can be used by the web browser on the PC side is only 1.5 to 2 GB, while loading Cesium requires almost 700 MB. The remaining memory for loading BIM models is very limited, so that even a slightly larger BIM model may cause the browser to crash.

1.2 The Computing Bottleneck of Web Browsers

In the traditional loading mode, BIM components are added one by one into the scene. It leads to a long initial loading time and low frame rate in the scene. In addition, the "WebGIS + WebBIM" platform not only needs to perform GIS rendering but also WebBIM visualization. The performance becomes worse due to the limited computing resources. The online editing functions cannot meet the real-time and interactive requirements of users under this circumstance.

In order to solve above problems, this paper discusses how to implement the Cesium based lightweight WebBIM visualization and proposes an improved plan for the lightweight processing of BIM data and lightweight WebBIM visualization based on Cesium, as well as a range of innovative editing functions.

The rest of this paper is organized as follows. Section 2 discusses the related works. Section 3 gives an introduction of key technologies. Section 4 introduces several novel WebBIM editing functions. Experimental results are given in Sect. 5 and Sect. 6 gives a conclusion of this work.

2 Related Work

To solve above bottleneck problems of Web Browsers, some related researches on WebGIS, WebBIM and online visualization technology are surveyed as follows.

2.1 WebGIS Based Smart City Visualization Management

In order to implement WebGIS part, the platform uses Cesium as the WebGIS engine. Cesium [14] is a cross-platform, cross-browser JavaScript library that displays 3D globes and maps. It uses WebGL [15] for hardware-accelerated graphics and does not require any plug-in support, but the browser must support WebGL. It is an open source program based on the Apache 2.0 license and is free for commercial and non-commercial use.

2.2 WebBIM Based Smart City Visualization Management

Considering the scale of the WebBIM models, the variety of the network and the limited cache of web browsers, most of the current BIM [1] applications are some of the primary BIM visualization applications.

The BIM [17] visualization mechanism based on WebGL (Three.js) is more popular among users and is expected to become the mainstream means of WebBIM visualization. Wang [6] used the open source IFC Optimizer [16] to perform semantic lightweight processing on BIM files, which effectively reduced the amount of data. Based on the WebGL framework Three.js, the BIM model was visualized on the webpage, and further the BIM information management system was realized [5]. The advantage of this method is that users can view WebBIM models directly on the web browser without installing the plug-in. However, the limited lightweight capabilities of IFC Optimizer will cause a serious load delay when rendering large-scale BIM models.

2.3 WebGIS and WebBIM Based Smart City Visualization Management

The “WebGIS + WebBIM” technology is still a new topic in the domestic BIM field. Cheng [3] proposed a BIM-GIS integration framework by integrating BIM into an ArcGIS-based GIS environment. This method still has the drawback of relying on Autodesk Revit as the BIM environment. Ma [4] proposed in the comprehensive application of BIM and GIS that extracting data from BIM system into GIS is the mainstream of BIM and GIS integration, but it needs to be improved in terms of building scale and visualization [2]. Fu [13] constructed a set of 3D BIM + GIS visual area digital reservoir security management system platform, but the browser side still needs to install plugins before viewing the model, which does not reflect the flexibility of B/S architecture, and the platform can only display one single BIM model at one time.

In summary, the current “WebGIS + WebBIM” visualization technology has the limitations of downloading plug-ins and weak rendering ability [7]. At the same time, although the HTML5/WebGL-based method solves the plug-in problems, it still lacks proper lightweight processing for large-scale BIM models [10].

Our contributions embody the following aspects: (1) Effective lightweight processing of BIM data; (2) Effective lightweight WebBIM virtualization technologies; (3) Successful integration of Cesium and Three.js; (4) Novel solutions for the management of smart city.

3 Key Technologies of Lightweight WebBIM

A technical roadmap based on all the required technologies in the platform is given. As shown in Fig. 1, the technical roadmap of this platform is mainly divided into the following three parts: lightweight processing of BIM data, lightweight visualization of WebBIM, and integration of Three.js with Cesium. Each module is described in detail in the following sections.

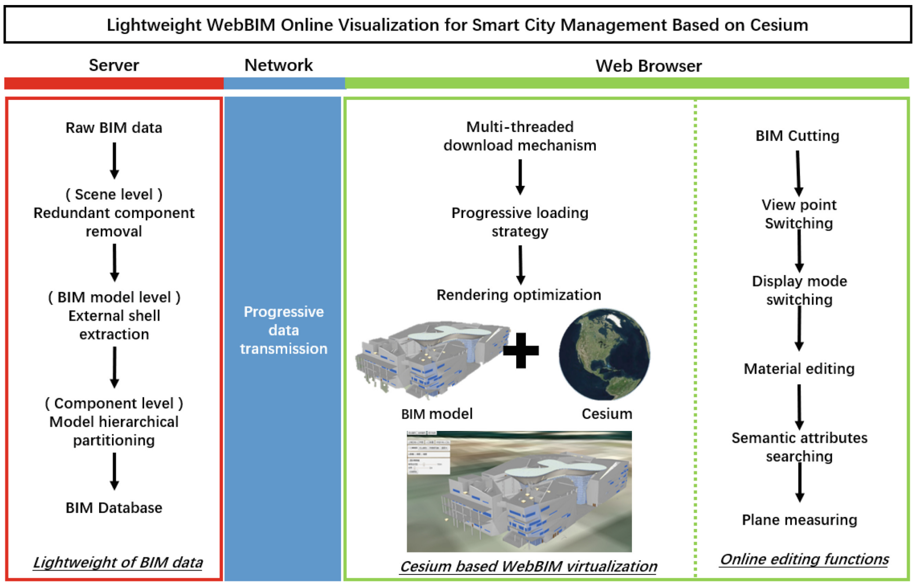
Lightweight WebBIM Online Visualization for Smart City Management Based on Cesium
Server
Network
Web Browser
Raw BIM data
(Scene level) Redundant component removal
(BIM model level) External shell extraction
(Component level) Model hierarchical partitioning
BIM Database
Lightweight of BIM data
Progressive data transmission
Multi-threaded download mechanism
Progressive loading strategy
Rendering optimization
BIM model
Cesium
Cesium based WebBIM virtualization
BIM Cutting
View point Switching
Display mode switching
Material editing
Semantic attributes searching
Plane measuring
Online editing functions

Fig. 1. Technical roadmap.

3.1 Lightweight Processing of BIM Data

Due to the limited memory space on the PC side, if the browser wants to load a larger BIM model, it must lightweight the BIM data first. The steps used are shown in Fig. 2.

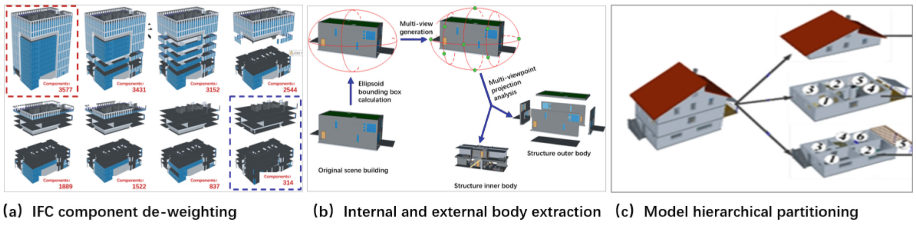

Fig. 2. Lightweight processing of BIM data.

The process in the above figure is divided into three steps for the lightweight processing of BIM data.

The first step is to perform geometric repeatability detection and redundant component removal on BIM models, reducing the number of components [8]. The second step forms a complete scene index according to the BIM structure and extracts the external shell of the BIM model. Due to the complexity of the inner BIM structure, even if the inner and outer parts are separated, it is impossible to render all the inner BIM components at one time. Therefore, the third step is to subspace the BIM inner scene based on the scene index and to find the closed subspace according to the adjacency relationship [9]. The BIM components are organized by the hierarchical index mechanism. In this way, the web browse only needs to load the corresponding subspace according to the actual position of the user in the scene.

All data preprocessed in the above three steps will be stored in the BIM database.

Next, the platform adaptively packages all BIM components using visual fullness (FFR) based prioritization [11]. The BIM components which have larger size and smaller amount of data are more easily to be seen by the camera's current perspective. So, they have a higher download priority. All BIM models in the same area with the same FFR priority will be packaged into a file for network transmission [12]. This method is used to batch-package a large number of files and finally output them into a number of priority-packed files. It makes the number of HTTP download request as small as possible and improve the efficiency of the network transmission.

3.2 Lightweight Online Visualization of WebBIM

The lightweight WebBIM online visualization technology mainly includes three parts: a multi-thread data download mechanism, a progressive loading strategy and an improvement on rendering optimization.

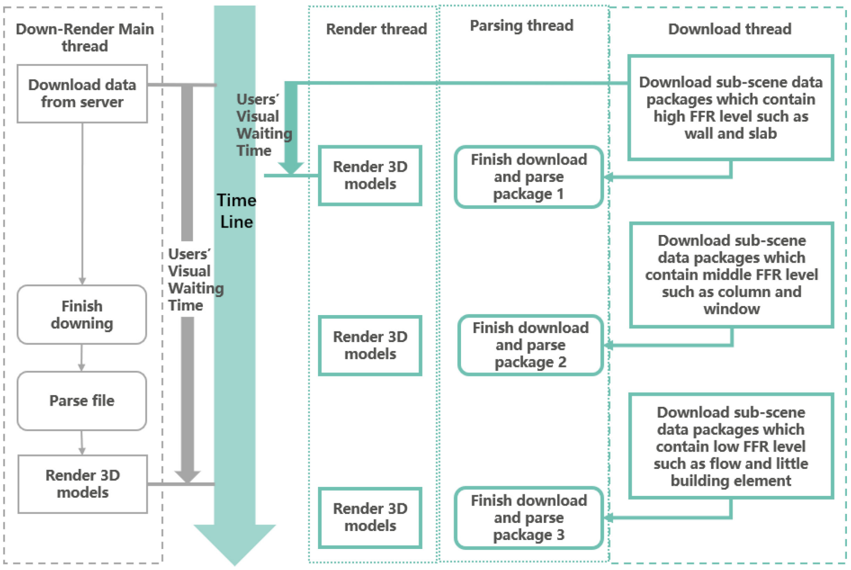
Down-Render Main thread
Download data from server
Finish downing
Parse file
Render 3D models
Users' Visual Waiting Time
Time Line
Users' Visual Waiting Time
Render thread
Render 3D models
Render 3D models
Render 3D models
Parsing thread
Finish download and parse package 1
Finish download and parse package 2
Finish download and parse package 3
Download thread
Download sub-scene data packages which contain high FFR level such as wall and slab
Download sub-scene data packages which contain middle FFR level such as column and window
Download sub-scene data packages which contain low FFR level such as flow and little building element

Fig. 3. Multi-threaded download rendering mechanism compared to the traditional way.

Multi-thread Data Download Mechanism. This paper proposes a three-thread download-parsing-rendering strategy to minimize the loading time and maximize the loading efficiency.

The rendering strategy comparison diagram is shown in Fig. 3. The left side is the traditional download-rendering process. This process only starts rendering after all the models have been downloaded, so the overall wait time can be very long. The right side represents a real-time rendering strategy based on multi-threading. When downloading data, the corresponding data packet is gradually downloaded according to their priority. Once a data packet is downloaded, the web browser will open a parsing thread different from the download thread. After parsing the downloaded data packet, the third thread is used to render BIM models immediately. In this way, users will have a shorter waiting time to view the BIM models.

Progressive Loading Strategy Based on Data Partitioning. When rendering indoor scenes, we design a region switching logic based on the connected graph obtained by the third step of lightweight processing of BIM data. The web browser deletes the existing model blocks before loading a new one. When loading inner scenes, the total amount of BIM data at the same time will not exceed the browser's memory threshold, thus ensuring that the browser can run normally. In this paper, a set of progressive loading strategy based on model partitioning is used to incrementally load model data through the position of the camera.

With progressive loading, the initial load time is reduced by approximately 90% compared to the time it takes to load all models into memory, and the memory consumption is significantly reduced compared with loading all models into the scene.

Rendering Optimization. Draw Call is used to command the GPU to perform rendering operations. Therefore, in order to optimize the rendering, we need to reduce the number of Draw Calls as much as possible. For this purpose, the Merge strategy based on the characteristics of Three.js is purposed, which combines multiple geometries into one geometry. The number of geometries in the scene is reduced significantly, from tens of thousands to just a few, thus ensuring the efficiency of rendering.

3.3 Integration of Three.js with Cesium

Since Three.js and Cesium are not uniform in the rendering principle and cannot share the same set of renderers, it is impossible to pass the Cesium rendered geometry to Three.js for editing or to directly submit the Three.js edited 3D object to Cesium for direct rendering. Considering that the two engines can't be coupled at the rendering level, this paper proposes a coupling mechanism based on geometric data sharing.

The whole system is dominated by Cesium. On the GIS system, the GIS-level model can be loaded and rendered. When the user needs to view the buildings on Cesium, they can click on the name of the building on the interactive menu and then the platform will download the external shell of the corresponding building and render it to the GIS system in time for there is no need to observe the details of the building on Cesium.

If the user needs to view or edit the BIM model, they can click on the specific BIM model and evoke the Three.js engine. Since the external shell of the BIM model has been downloaded in Cesium, the outline of the building will be presented immediately. At the same time, in the WebBIM mode, the downloaded data is further loaded into the memory and the rendering is selectively performed according to the user's position. When the user closes the WebBIM mode or switches the browsing object, all geometric data of the building is released from the memory to ensure that the memory consumption is as small as possible.

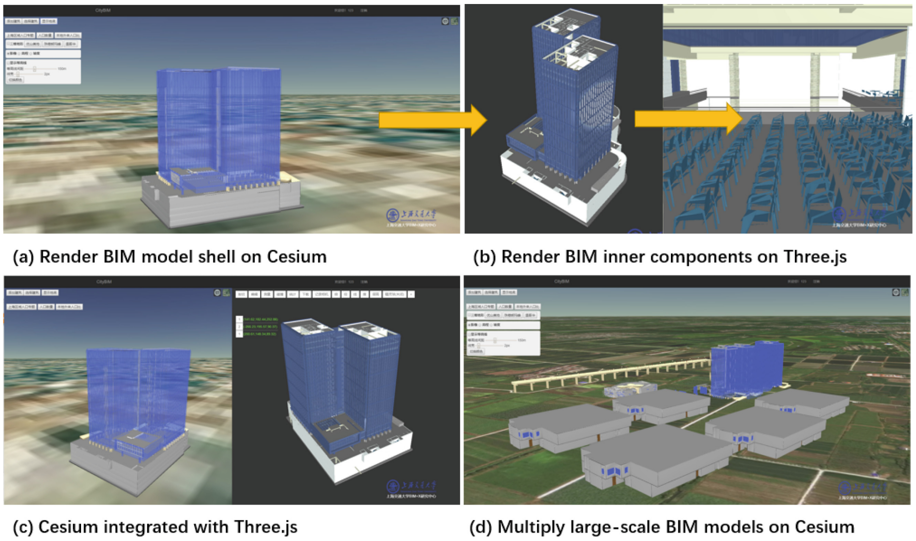
(a) Render BIM model shell on Cesium
(b) Render BIM inner components on Three.js
(c) Cesium integrated with Three.js
(d) Multiply large-scale BIM models on Cesium

Fig. 4. Model rendering based on Cesium and Three.js.

As the Fig. 4 shows, (a) displays rendering a BIM model shell on Cesium. (b) shows how to render BIM inner components using Three.js. (c) displays the operation interface of the platform, with Cesium part on the left and WebBIM part on the right. (d) renders multiply large-scale BIM model on Cesium at the same time.

4 Online Editing Functions

After implementing the lightweight processing of BIM data and WebBIM virtualization, the actual project requirements and WebCityBIM platform features led to the development of many online editing functions. In this section, the following two functions are described in detail, and the rest are given by the brief introductions.

4.1 Scene Cutting Based on Spatial Analysis

Scene cutting can display the internal structure of BIM models and is one of the key applications of the smart city management platform. Since the entire scene is managed according to voxelization, the inner components are not loaded when the external shell is cut. When the scene cutting is performed, the position of the cutting plane is calculated and converted into a voxel index. According to the voxel index, the voxel coordinates in the plane are found. Then the corresponding inner components are found according to the voxel coordinates and immediately rendered into the scene. A schematic diagram of the scene cutting from multiple angles in the scene is shown in Fig. 5(a).

4.2 View Point Switching

The view point switching function can save camera view points when the user browses the BIM models. When the user clicks on the record camera button and enters a new camera position. The system will save it into the database together with the current camera view point, which is displayed on the left side of the operation interface. After the user clicks the button representing the camera position label, the database reads the corresponding camera position coordinates from the database, and gradually moves from the current position to the selected position.

As is shown in Fig. 5(b), the view point switching function is a gradual process, and the model generates translation and rotation operations during the movement.

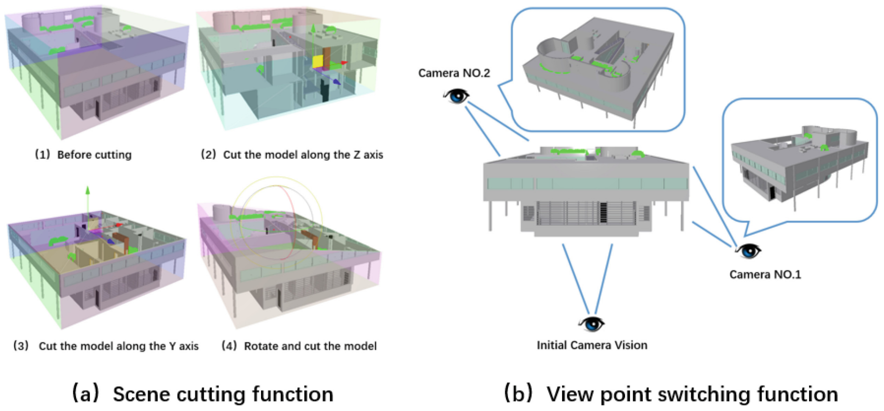
(1) Before cutting
(2) Cut the model along the Z axis
(3) Cut the model along the Y axis
(4) Rotate and cut the model
(a) Scene cutting function
Camera NO.2
Camera NO.1
Initial Camera Vision
(b) View point switching function

Fig. 5. Scene cutting and perspective storage function.

4.3 Other Online Editing Functions

Some other editing functions are discussed below briefly.

- Display mode switching: Display BIM models between entity map and line block diagram.
- Semantic attributes searching: Support the semantic attributes query and the graphic annotation of the models.
- Material editing: Support the material replacement effect of the model and the function preview.
- Plane measuring: Add a plane to the scene and the plane can be translated, stretched and rotated. The user can add measurement points to the plane by clicking on the plane and calculating the distance between the points.

- Photogrammetric model loading: The photogrammetric model is lightened and converted to GLTF format using a format conversion tool and then rendered on the Cesium platform.

The operation diagram of each function is shown in Fig. 6.

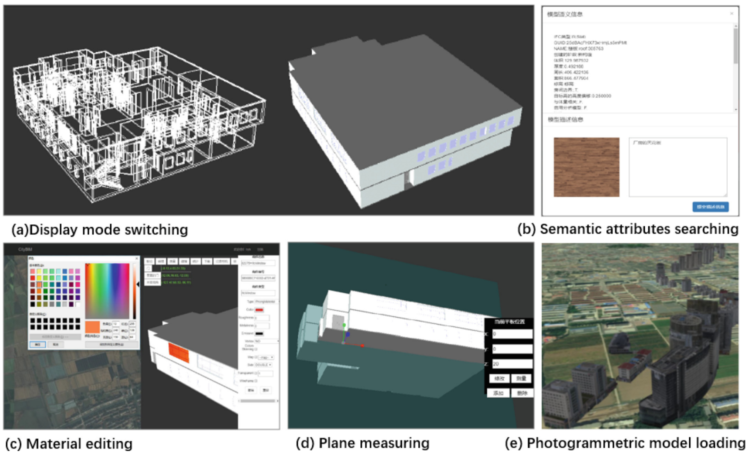
(a)Display mode switching
(b) Semantic attributes searching
(c) Material editing
(d) Plane measuring
(e) Photogrammetric model loading

Fig. 6. Platform function operation diagram.

5 Experimental Results and Performance Analysis

According to the performance evaluation method of Web3D platform, this paper proposes three performance indicators for evaluating experimental results: the initial loading time of the scene; the memory occupancy rate after loading the BIM model; and the frame rate of the browser. The test content mainly includes two sets of test cases: the efficiency test of progressive loading, the performance test of Cesium combined with Three.js engine.

5.1 Test Environment

Desktop and laptop browser are used for testing. Common configuration is listed as Server OS: WinServer2008 Mem: 4G, CPU: Intel Xeon 2.39G, Client: OS: Windows7 Mem: 8G, CPU: Intel i7-4700MQ 100 units, Browser: Google Chrome 52.0.2743, Network bandwidth: 100Mbps laboratory, campus outlet bandwidth 4.5Gbps.

5.2 Test Case

The static scene data used in this test is BIM model data with the base format of IFC. Take the twin towers model in Fig. 4 as an example. The geometric information data volume is 1.2G, and the total number of triangular polygons is 18,850,000.

5.3 The Efficiency Test of Progressive Loading

As one of the core technologies of the platform, progressive loading determines whether the web engine can load and render the scenes that users need to see quickly and efficiently. In this test, we compare the initial loading time of the scene, the memory usage after loading, and the overall rendering efficiency with progressive loading and without progressive loading when loading static large-scale scenes.

Test Result. With progressive loading, the initial load time is reduced by approximately 90% compared to the time it takes to load all models into memory. Moreover, progressive loading is significantly reduced in memory consumption relative to all model loads as is shown in Fig. 7.

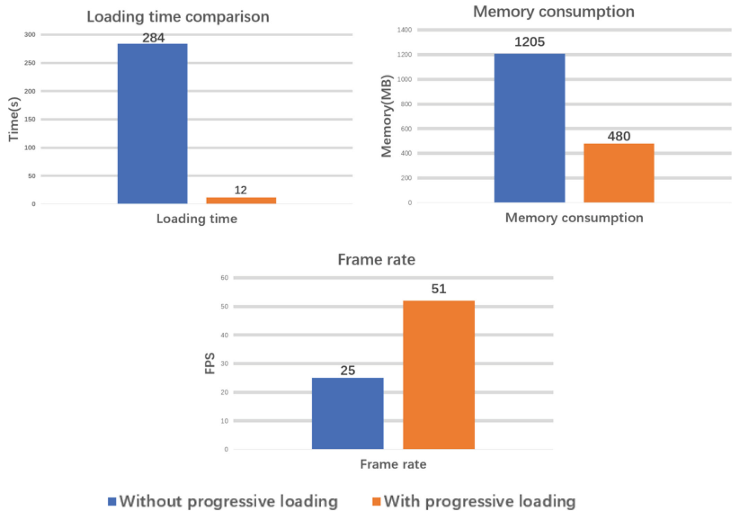

Fig. 7. Progressive loading performance comparison.

5.4 The Performance Test of Cesium Combined with Three.js Engine

The second core technical point of this paper is how to integrate Cesium and Three.js. The key technical difficulty lies in how to load Three.js model and guarantee rendering performance based on Cesium. So, the second test focused on the evaluation of all aspects of the performance of the engine after the platform was combined.

Test Result. Through the test, we can see the experimental data shown in the following Table 1. When the model is not loaded, due to the characteristics of the Cesium platform, a certain amount of memory is occupied. On this basis, through our lightweight methods, when loading large-scale BIM models, the growth of memory will not be large and will be stable within the threshold. At the same time, opening the Three.js engine module for editing will also ensure that the memory will not crash. The rendering frame rate can also be maintained above 20 frames. As is shown in Fig. 8, the system is proved to be effective in Cesium based lightweight WebBIM virtualization.

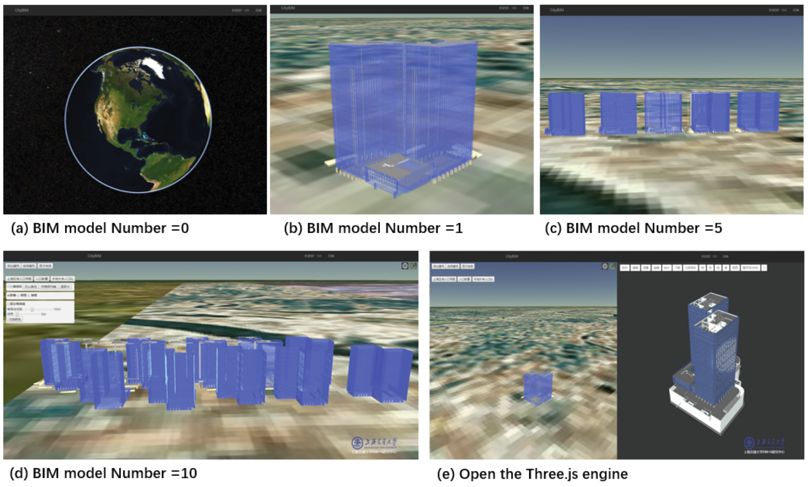

Fig. 8. Performance test.

Table 1. Performance test result.

Number of buildings	0	1	5	10
Three.js engine rendering	OFF	OFF/ON	OFF/ON	OFF/ON
Memory consumption	690 MB	710 MB/1.1 GB	750 MB/1.12 GB	920 MB/1.3 GB
Rendering frame rate	55FPS	51FPS/45FPS	42FPS/37FPS	36FPS/28FPS

6 Conclusion and Future Work

This paper proposes the Cesium based lightweight WebBIM technology for smart city virtualization management, dealing with large amount of BIM data and the rendering problems. The main work is divided into three parts: lightweight processing of BIM data, lightweight virtualization of WebBIM models and the integration of Three.js and Ceisum. The platform successfully combines the Three.js engine with the Ceisum platform to realize online interaction of terrain information and BIM information, also provides a variety of novel functions to support the practical requirements of the smart city management.

In conclusion, this platform is still in its infancy for the lightweight of large-scale BIM scenarios, and there are still many aspects to be improved. The next phase of the work plan includes the use of binary database storage to lightweight BIM data and the use of LOD multi-level detailing techniques to render larger-scale urban-level Web-BIM scenarios on Cesium.

References

1. Chuang, T.H., Lee, B.C., Wu, I.C.: Applying cloud computing technology to BIM visualization and manipulation. In: International Symposium on Automation and Robotics in Construction, pp. 144–149 (2011)
2. Ma, Z., Liu, Z.: BIM-based intelligent acquisition of construction information for cost estimation of building projects. Procedia Eng. **85**, 358–367 (2014)
3. Cheng, C.P., Deng, Y.: An integrated BIM-GIS framework for utility information management and analyses (2015)
4. Ma, Z., Ren, Y.: Integrated application of BIM and GIS: an overview. Procedia Eng. **196**, 1072–1079 (2017)
5. Zhang, J., Yu, F., Li, D., et al.: Development and implementation of an IFC based graphic information model for virtual construction. Comput. Aided Civil Infrastruct. Eng. **29**(1), 60–74 (2014)
6. Wang, M.F., Jia, J.Y., Xie, N., Zhang, C.X.: Interest-driven avatar neighbor-organizing for P2P transmission in distributed virtual worlds. In: Computer Animation & Virtual Worlds (2015)
7. Das, M., Cheng, C.P., Kumar, S.S.: Social BIM-cloud: a distributed cloud-based BIM platform for object-based lifecycle information exchange. Vis. Eng. **3**(1), 1–20 (2015)
8. Liu, X.J., Xie, N., Jia, J.Y.: WebVis_BIM: real time Web3D visualization of big BIM data. In: ACM SIGGRAPH International Conference on Virtual Reality Continuum and its Applications in Industry, pp. 43–50 (2015)
9. Zhou, W., Tang, K., Jia, J.Y.: S-LPM: segmentation augmented light-weighting and progressive meshing for the interactive visualization of large man-made Web3D models. World Wide Web J. (WWWJ) **21**(5), 1425–1448 (2018)
10. Wang, H.W., Hu, Z.Z., Lin, J.R., Zhang, J.P.: Web-oriented BIM 3D viewing and information management. J. Inf. Technol. Civil Eng. Archit. **3** (2013)
11. Liu, X.J., Jia, J.Y.: Mobile web-based lightweight and real-time roaming algorithm for large-scale WebBIM scenes. Scientia Sinica (Informations) **48**(03), 274–292 (2018)
12. Hu, Y.H., Chen, C.H., Liu, X.J., Huang, F., Jia, J.Y.: WebTorrent based fine-grained P2P transmission of large-scale WebVR indoor scenes. In: ACM Web3D, Brisbane, Australia (2017)
13. Fu, S.Y., Zhao, Z.Y., Yang, S.W., Fang, Z.Y., Fang, W.: Modeling of digital regional reservoir based on 3D BIM and WebGIS. J. Yangtze River Sci. Res. Inst. **35**(04), 134–136 +142 (2018)
14. Cesium. https://cesiumjs.org/. Accessed 22 Mar 2019
15. WebGL Engine. http://www.threejs.org. Accessed 20 Mar 2019
16. Solibri IFC Optimizer. http://www.solibri.com/. Accessed 19 Mar 2019
17. BIM server. http://www.bimserver.org. Accessed 20 Mar 2019